DOI: 10.15514/ISPRAS–2020–32(6)–2

Верификация соответствия между разноуровневыми моделями функциональных требований

А.В. Хорошилов, ORCID: 0000-0002-6512-4632 <khoroshilov@ispras.ru>

Институт системного программирования им. В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
Московский государственный университет имени М.В. Ломоносова,
119991, Россия, Москва, Ленинские горы, д. 1
НИУ Высшая школа экономики,
101978, Россия, г. Москва, ул. Мясницкая, д. 20
Московский физико-технический институт,
141701, Россия, Московская область, г. Долгопрудный, Институтский пер., 9

Аннотация. В статье предлагаются методы доказательства соответствия между разноуровневыми моделями, нацеленные на доказательство свойств безопасности в индивидуальной трассовой семантике. Эти методы более просты и пригодны для применения при решении практических задач верификации сложных видов функциональных требований по сравнению с традиционными подходами, основанными на установлении отношения уточнения между моделями.

Ключевые слова: формальная верификация; формальные модели; трассовая семантика.

Для цитирования: Хорошилов А.В. Верификация соответствия между разноуровневыми моделями функциональных требований. Труды ИСП РАН, том 32, вып. 6, 2020 г., стр. 19-30. DOI: 10.15514/ISPRAS–2020–32(6)–2

Благодарности. Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-07-00954.

Verification of Compliance for Multilevel Models in Individual Trace Semantics

A.V. Khoroshilov, ORCID: 0000-0002-6512-4632 <khoroshilov@ispras.ru>

Ivannikov Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia,
Lomonosov Moscow State University,
GSP-1, Leninskie Gory, Moscow, 119991, Russia
National Research University, Higher School of Economics
20, Myasnitskaya Ulitsa, Moscow, 101978, Russia
Moscow Institute of Physics and Technology (MIPT)
141700, Russia, Moscow region, Dolgoprudny, Campus per., 9

Abstract. The paper considers the problem of verification of compliance between models representing the same system on different level of abstraction. The existing approaches are mostly based on refinement relation. But the models representing industrial systems are quite big and complex, while semantics gap between the level is

quite big. As a result, the existing methods became too complex and labour intensive. The paper presents new verification techniques that targets to prove multimodel compliance in terms of individual trace semantics. The techniques assume that each model is verified, i.e. it is proved that starting from initial states of labelled transition system is not possible to reach unsafe states by using valid transitions. The first proposed technique allows to prove that the detailed model satisfies to requirements of the abstract model, i.e. reachable states of detailed model do not include states corresponding to unsafe states of the abstract model. The second proposed technique allows to prove that the detailed model satisfies to behaviour specification of the abstract model, i.e. all reachable transitions of the detailed model do not include transitions corresponding to invalid transitions of the abstract model. For each technique the correspondence relation is defined in terms of the models, i.e. the relations are formally defined and they can be used for analysis with interactive or automated provers. At the same time, there are some requirements to that relations that are expressed in terms of low level events that exist hypothetically only and can be analyzed theoretically only. As a result, the proposed techniques provides a reasonable approach to prove compliance between mulilevel models in more approachable way for industrial settings.

Keywords: formal verification; formal models; trace semantics

For citation: Khoroshilov A.V. Verification of compliance for multilevel models in individual trace semantics. Trudy ISP RAN/Proc. ISP RAS, vol. 1, issue 2, 2017. pp. 19-30 (in Russian). DOI: 10.15514/ISPRAS-2020-32(6)-2

Acknowledgements. The reported study was funded by RFBR, project number 20-07-00954.

1. Введение

Формальные модели программных систем являются одним из механизмов повышения доверия к безопасности и надёжности программных систем, решающих ответственные задачи, такие как защита информацией и управление доступом в информационно-вычислительных системах в условиях враждебной среды, управление ответственным оборудованием и т.д. Во многих случаях для одной программной системы разрабатывается несколько формальных моделей, описывающих разные аспекты её функционирования на разных уровнях детализации. Эти модели при описании практически значимых программных систем являются достаточно большими и сложными, а семантический разрыв между моделями разных уровней оказывается достаточно большой, чтобы была возможность эффективно применять существующие методы верификации соответствия между разноуровневыми моделями. Основная причина этого заключается в объективной сложности данной задачи в её полной постановке.

В рамках настоящей работы предлагаются методы доказательства соответствия между разноуровневыми моделями, нацеленные на доказательство свойств безопасности в индивидуальной трассовой семантике. Эти методы более просты и пригодны для применения при решении практических задач верификации сложных видов функциональных требований по сравнению с традиционными подходами, основанными на установлении отношения уточнения между моделями.

Дальнейшее изложение материала организовано следующим образом. В разд. 2 вводятся базовые термины, используемые далее. Разд. 3 содержит описание предлагаемых методов и доказательства их корректности. В разд. 4 обсуждаются вопросы практического применения методов и направления дальнейших исследований. В заключении приводятся краткие выводы по результатам работы.

2. Предварительные определения

В рамках данной работы целевой системой мы будем называть программную или программно-аппаратную систему, которая подлежит моделированию и верификации.

2.1 Формализация поведения целевой системы

Прежде чем рассматривать модели целевой системы, с которыми выполняется непосредственная работа по анализу и верификации, введём понятие низкоуровневой модели поведения целевой системы, которая предоставит аппарат для дальнейших рассуждений о функциональных свойствах её поведения.

Определение 1. Обозначим **LLEvents** множество всех потенциально возможных низкоуровневых событий, из который состоит функционирование целевой системы. Состав событий может зависеть от природы целевой системы. Например, для операционных систем такими событиями могут быть события аппаратного уровня, такие как изменение значения регистров, поступление прерываний и т.д. Предполагается, что элементы этого множества представляют собой не только тип события, но все его сопутствующие атрибуты, включая его пространственные и временные характеристики, поэтому без ограничения общности будем считать, что любые два возможных события в аппаратуре представляются различными элементами **LLEvents**. Также будем считать, что на множестве **LLEvents** задано антирефлексивное бинарное отношение $\prec_{dep}$, которое означает, что второе событие причинно зависит от первого события явным или неявным образом, в частности, второе событие может произойти по времени только строго позже первого.

Определение 2. Трассой низкоуровневых событий будем называть конечное или бесконечное финитарное частично-упорядоченное множество событий $(\mathbf{T}, \preccurlyeq)$, где

- $\mathbf{T} \subseteq \mathbf{LLEvents}$;
- частичный порядок $\preccurlyeq$ является транзитивным и рефлексивным замыканием $\prec_{dep}$, т.е. $\preccurlyeq = \left(\{(e,e) | e \in \mathbf{T}\} \cup \prec_{dep|T}\right)^*$ и определяет порядок событий во времени и их причинную зависимость;
- $\prec_{dep|T} = \prec_{dep} \cap\ \mathbf{T} \times \mathbf{T}$ — ограничение отношения $\prec_{dep}$ на множество **T**;
- свойство финитарности означает, что $\forall\, e \in \mathbf{T}$ множество $\{e' \in \mathbf{T} \mid e' \preccurlyeq e\}$ является конечным.

Далее будем использовать следующие обозначения:

- $\wp(\mathbf{X})$ – множество всех подмножеств множества **X**;
- $\wp_{\text{fpo}}(\mathbf{X})$ – множество всех финитарных частично-упорядоченных подмножеств множества **X**;
- $\mathbf{Traces}(\mathbf{LLEvents}, \prec_{dep}) \subseteq \wp_{\text{fpo}}(\mathbf{LLEvents})$ – множество всех трасс событий над множеством **LLEvents** с отношением зависимости $\prec_{dep}$;
- $\wp_{\text{directed}}(\mathbf{X}, \preccurlyeq) = \{\mathbf{X}' \subseteq \mathbf{X} \mid \forall\, \mathbf{T}_1, \mathbf{T}_2 \in \mathbf{X}'\ \exists\, \mathbf{T}_3 \in \mathbf{X}' \colon \mathbf{T}_1 \preccurlyeq \mathbf{T}_3 \wedge \mathbf{T}_2 \preccurlyeq \mathbf{T}_3)\}$ – множество всех направленных подмножеств частично-упорядоченного множества **X**;
- $\wp_{\text{dcpo}}(\mathbf{X}, \preccurlyeq) = \{\mathbf{X}' \in \wp(\mathbf{X}) \mid \forall\, \mathbf{Y} \in \wp_{\text{directed}}(\mathbf{X}', \preccurlyeq|_{\mathbf{X}'})\ \mathbf{sup}(\mathbf{Y}) \in \mathbf{X}'\}$ – множество всех dcpo-подмножеств множества **X**, т.е. подмножеств, являющихся направленными полными частично-упорядоченными множествами;
- $\mathbf{Prefix}(\mathbf{T}, \preccurlyeq) = \{(\mathbf{T}', \preccurlyeq') | \mathbf{T}' \subseteq \mathbf{T} \wedge \forall e \in \mathbf{T}\ \forall e' \in \mathbf{T}'\ (e \preccurlyeq e') \Rightarrow e \in \mathbf{T}' \wedge \preccurlyeq' = \preccurlyeq|_{\mathbf{T}'}\}$ – множество всех префиксов частично-упорядоченного множества $(\mathbf{T}, \preccurlyeq)$, а $\mathbf{Prefix}^{\text{fin}}(\mathbf{T}, \preccurlyeq)$ – множество всех конечных префиксов;
- $\preccurlyeq_{\text{pre}} = \{(\mathbf{T}_1, \mathbf{T}_2) \mid \mathbf{T}_1 \in \mathbf{Prefix}(\mathbf{T}_2)\}$ – отношение «является префиксом» на множестве частично-упорядоченных множеств;
- $\wp_{\text{cleft}}(\mathbf{X}) = \left\{\mathbf{X}' \in \wp_{\text{dcpo}}\left(\mathbf{X}, \preccurlyeq_{\text{pre}}\right) \middle| \forall (\mathbf{T}, \preccurlyeq) \in \mathbf{X}'\ \mathbf{Prefix}(\mathbf{T}, \preccurlyeq) \subseteq \mathbf{X}'\right\}$ — множество всех **dcpo**-подмножеств **X** относительно порядка $\preccurlyeq_{\text{pre}}$, являющихся замкнутыми влево относительно частичных порядков своих элементов, где **X** – некоторое множество частично-упорядоченных множеств.

Определение 3. Обозначим $\mathbf{LLTraces} \in \wp_{\text{cleft}}(\mathbf{Traces}(\mathbf{LLEvents}, \prec_{dep}))$ множество потенциально возможных трасс низкоуровневых событий. Требование потенциальной возможности трассы подразумевает, что трасса является согласованной с правилами корректного поведения окружения целевой системы, то есть, например, в ней отсутствуют подмножества событий, возможные только в случае сбоев в аппаратуре.

В рамках данных определений любая целевая система характеризуется элементом множества $\wp_{\text{cleft}}(\mathbf{LLTraces})$, обозначаемым **LLModel**, который представляет собой множество трасс, в которых целевая система может участвовать при всех возможных сценариях развития событий.

Любое требование к целевой системе может быть представлено в виде подмножества $\wp_{\text{cleft}}(\mathbf{LLTraces})$, то есть множества множеств трасс, которые являются удовлетворяющими данному требованию.

В рамках данной работы мы ограничимся рассмотрением требований безопасности в индивидуальной трассовой семантике [1] (т. е. требований подкласса $\mathbf{S_1 L_0}$ класса $\mathbf{HN}_1$ [2]), которые задаются множеством корректных трасс $\mathbf{LLSafe} \subseteq \mathbf{LLTraces}$, таким что $\forall\, t \in \mathbf{LLTraces}\ t \notin LLSafe \Rightarrow \exists t_m \in \mathbf{Prefix}^{\text{fin}}(t) \colon \forall\, t' \in \mathbf{LLTraces}\ t_m \in \mathbf{Prefix}^{\text{fin}}(t') \Rightarrow t' \notin \mathbf{LLSafe}$. Другими словами, целевая система удовлетворяет такому требованию, если **LLModel** является подмножеством **LLSafe**, а множество **LLSafe** характеризуется тем, что оно запрещает наличия в трассе «плохих событий», то есть появление такого события относит трассу к числу недопустимых.

Класс требований безопасности достаточно широк, например, в него входят требования, которые выражаются в виде предусловий и постусловий операций, а также требования отсутствия в трассе недопустимых событий. Среди требований, которые не входят в этот класс, можно привести, например, следующие:

- требования живучести, то есть требования, которые предписывают бескрайнего появления «хороших событий»;
- требования существования, то есть требующие наличия среди всех возможных трасс выполнения трасс с определёнными свойствами;
- требования детерминизма, например, требующие, чтобы определённая функция в определённых условиях возвращала одно из множества допустимых значений, но для фиксированной целевой системы это значение всегда было одним и тем же.

2.2 Абстрактные модели целевой системы

Поскольку низкоуровневая модель поведения целевой системы слишком детальна и сложна, то на практике для верификации используются более абстрактные модели. Одним из наиболее распространённых механизмов описания таких моделей являются автоматные модели, которые для нашего рассмотрения будут представляться как системы размеченных переходов.

Определение 4. Система размеченных переходов **LTS** — это четвёрка $(S, S_{\text{init}}, L, \rightarrow)$, где

- S – множество состояний;
- $S_{\text{init}} \subseteq S$ – непустое множество начальных состояний;
- L – множество меток;
- $\rightarrow \subseteq S \times L \times S$ – множество переходов между состояниями, помеченных метками.

Определение 5. Трассой **LTS** называется конечная или бесконечная последовательность переходов, такая что

- если последовательность не пустая, то начальное состояние первого перехода принадлежит S_{init},
- начальное состояние i-го перехода в последовательности совпадает с конечным состоянием $(i-1)$-го перехода для $i > 1$.

Множество всех возможных трасс системы размеченных переходов **LTS** будем обозначать $\mathbf{LTSTraces}(\mathbf{LTS})$. Когда из контекста понятно о какой системе размеченных переходов идёт речь, будет использоваться сокращённое обозначение **LTSTraces**. Множество состояний, входящих в переходы из трассы **T**, будем обозначать $\mathbf{states}(\mathbf{T})$, а множество конечных состояний переходов из трассы **T**, будем обозначать $\mathbf{endstates}(\mathbf{T})$.

При построении **LTS** для моделирования поведения целевой системы любой трассе низкоуровневых событий $\mathbf{T}_{\mathrm{LL}} \in \mathbf{LLTraces}$ соответствует $\alpha(\mathbf{T}_{\mathrm{LL}}) \in \mathbf{LTSTraces}$, которое представляет собой трассу, задающую модельное поведение, представленное в терминах модели **LTS**. При этом для $\alpha(\mathbf{T}_{\mathrm{LL}})$ требуется, чтобы $\forall\, \mathbf{T'}_{\mathrm{LL}} \in \mathbf{Prefix}(\mathbf{T}_{\mathrm{LL}})\ \alpha(\mathbf{T'}_{\mathrm{LL}})$ являлась префиксом последовательности $\alpha(\mathbf{T}_{\mathrm{LL}})$. Также предполагается, что конечным трассам низкоуровневых событий **TLL** соответствуют конечные трассы абстрактных событий $\alpha(\mathbf{T}_{\mathrm{LL}})$.

Иногда при определении функции α поддерживается формирование специальных значений, обозначающих невозможность построения абстракции для данной низкоуровневой трассы. В рамках данной работы для упрощения изложения считается, что при необходимости в абстрактной модели присутствуют элементы, явным образом моделирующие такие ситуации.

LTS, как правило, описывается на том или ином языке спецификаций, а отображение α остаётся в голове у автора спецификации и используется им для оценки адекватности разработанной **LTS**. Поэтому хорошей практикой является стараться, чтобы α была устроена достаточно прямолинейно, чтобы избежать ошибок в формулировке **LTS**, которые не могут быть выявлены инструментами из-за отсутствия формальной модели низкого уровня.

Разработка модели **LTS** проводится с двумя основными целями:

- получения чёткой, однозначно интерпретируемой формулировки модели поведения целевой системы на заданном уровне абстракции и выявления за счёт этого целого ряда проблем, связанных с внутренними противоречиями и неоднозначностями в понимании требований к системе [3];
- доказательство корректности модели поведения относительно требований, заданных в терминах той же модели **LTS**.

Для выражения требований к целевой системе могут быть использованы, например, формулы в той или иной темпоральной логике, использующей **LTS** в качестве основы для означивания пропозициональных переменных. Другой вариант заключается в описании требований в виде инвариантов на состоянии модели **LTS**. Далее рассмотрим подробнее второй вариант, поскольку он чаще возникает в нашей практике верификации функциональных требований к системам защиты информации.

Итак, для целей верификации дополнительно к базовой модели **LTS** добавляются два элемента $(S_{\mathrm{safe}}, \rightarrow_{\mathrm{safe}})$, где

- $S_{\mathrm{safe}} \subseteq S$ – множество безопасных состояний;
- $\rightarrow_{\mathrm{safe}} \subseteq \rightarrow$ – множество корректных переходов.

В качестве цели для доказательства формулируется утверждение, что множество достижимых состояний из начальных состояний S_{init} по переходам из множества $\rightarrow_{\mathrm{safe}}$ невозможно попасть в небезопасное состояние из $S_{\mathrm{unsafe}} = S \setminus S_{\mathrm{safe}}$. Доказательство, как правило, проводится по индукции через утверждения вида $S_{init} \subseteq S_{\mathrm{safe}}$ и $\rightarrow_{\mathrm{safe}} [S_{\mathrm{safe}}] \subseteq S_{\mathrm{safe}}$, то есть что любое начальное состояние является безопасным и любой корректный переход из безопасного состояния ведёт в безопасное состояние.

В терминах низкоуровневых поведений модель поведения корректно реализованной целевой системы задаётся при помощи множества корректных трасс:

$$\mathbf{LLSafeT}_{\mathrm{LTS}} = \mathbf{LLTraces} \setminus$$
$$\{\mathbf{T}_{\mathrm{LL}} \in \mathbf{LLTraces} \mid \exists\ \mathbf{T'}_{\mathrm{LL}} \in \mathbf{Prefix}^{\mathrm{fin}}(\mathbf{T}_{\mathrm{LL}}) \colon \alpha(\mathbf{T'}_{\mathrm{LL}}) \cap \rightarrow_{\mathrm{unsafe}} \neq \emptyset\},$$

где $\rightarrow_{\mathrm{unsafe}} = \rightarrow \setminus \rightarrow_{\mathrm{safe}}$, а запись $\alpha(\mathbf{T'}_{\mathrm{LL}}) \cap \rightarrow_{\mathrm{unsafe}} \neq \emptyset$ обозначает отсутствие некорректных переходов в трассе $\alpha(\mathbf{T'}_{\mathrm{LL}})$.

Соответственно, доказываемое требование задаётся при помощи множества корректных трасс:

$$\mathbf{LLSafeS}_{\mathrm{LTS}} = \mathbf{LLTraces} \setminus$$
$$\{\mathbf{T}_{\mathrm{LL}} \in \mathbf{LLTraces} \mid \exists\ \mathbf{T'}_{\mathrm{LL}} \in \mathbf{Prefix}^{\mathrm{fin}}(\mathbf{T}_{\mathrm{LL}}) \colon \mathbf{states}(\alpha(\mathbf{T'}_{\mathrm{LL}})) \cap S_{\mathrm{unsafe}} \neq \emptyset\}.$$

Таким образом, результатом верификации **LTS** модели является следующее утверждение: Если целевая система удовлетворяет требованию $\mathbf{LLSafeT}_{\mathrm{LTS}}$, то она удовлетворяет и требованию $\mathbf{LLSafeS}_{\mathrm{LTS}}$, или $\mathbf{LLSafeT}_{\mathrm{LTS}} \subseteq \mathbf{LLSafeS}_{\mathrm{LTS}}$.

3. Метод доказательства соответствия между разноуровневыми моделями

Достаточно распространённой ситуацией является появление нескольких абстрактных моделей, описывающих целевую систему на разных уровнях детализации. При этом возникает естественное желание доказать, что более детальная модель удовлетворяет требованиям более абстрактной. Традиционный подход к решению данной задачи заключается в использовании отношения уточнения между моделями [4], включая различные вариации, допускающие трансформацию алфавита моделей [5], введение дополнительных действий на детальном уровне [6, 7], а также разбиение абстрактных действий на множество детальных [8, 9], в том числе в более сложных соотношениях (*n* к *m*) [10, 11].

Тем не менее, на практике встречаются случаи, где применение даже наиболее гибких методов установления отношения уточнения оказывается неприемлемо сложным и трудоёмким, поэтому возникает потребность в более эффективных методах установления соответствия между разноуровневыми моделями.

3.1 Доказательство соответствия свойствам безопасности

Рассмотрим, как эта проблема может быть рассмотрена с формальной точки зрения в терминах трассовой семантики.

Предположим, что более абстрактная модель целевой системы задана системой размеченных переходов **LTS1** $(S_1, S_{\mathrm{init1}}, L_1, \rightarrow_1)$, множеством безопасных состояний S_{safe1} и множеством корректных переходов $\rightarrow_{\mathrm{safe1}}$, а более детальная модель – системой размеченных переходов **LTS2** $(S_2, S_{\mathrm{init2}}, L_2, \rightarrow_2)$, множеством безопасных состояний S_{safe2} и множеством корректных переходов $\rightarrow_{\mathrm{safe2}}$.

Тогда первое желаемое свойство для доказательства соответствия между моделями заключается в том, чтобы целевые системы, описываемые более детальной моделью, удовлетворяли свойствам безопасности, доказанным для абстрактной модели. Формально это можно сформулировать так: Если целевая система удовлетворяет требованию $\mathbf{LLSafeT}_{\mathrm{LTS2}}$, то она удовлетворяет и требованию $\mathbf{LLSafeS}_{\mathrm{LTS1}}$ или $\mathbf{LLSafeT}_{\mathrm{LTS2}} \subseteq \mathbf{LLSafeS}_{\mathrm{LTS1}}$.

Определение 6. Для $\mathbf{T}_{\mathrm{LL}} \in \mathbf{LLTraces}^{\mathrm{fin}}$ обозначим $\mathbf{newtrace}(\mathbf{T}_{\mathrm{LL}}, \alpha)$ хвост трассы $\alpha(\mathbf{T}_{\mathrm{LL}})$, такой что он не появлялся ни в одной трассе $\alpha(\mathbf{T'}_{\mathrm{LL}})$ ни для одного $\mathbf{T'}_{\mathrm{LL}} \in \mathbf{Prefix}(\mathbf{T}_{\mathrm{LL}})$. Соответственно, обозначим $\mathbf{newstates}(\mathbf{T}_{\mathrm{LL}}, \alpha)$ множество $\mathbf{endstates}(\mathbf{newtrace}(\mathbf{T}_{\mathrm{LL}}, \alpha))$, если $\mathbf{newtrace}(\mathbf{T}_{\mathrm{LL}}, \alpha)$ не пусто, и пустое множество в противном случае.

Определение 7. Отношением соответствия между состояниями моделей **LTS1** и **LTS2** будем называть отношение $\mathbf{R} \subseteq S_1 \times S_2$, которое удовлетворяет следующему условию:

$$\forall\, \mathbf{T}_{\mathrm{LL}} \in \mathbf{LLTraces}^{\mathrm{fin}}\ \forall\, s_1 \in \mathbf{newtrace}(\mathbf{T}_{\mathrm{LL}}, \alpha_1)\ \exists\, s_2 \in \mathbf{states}(\alpha_2(\mathbf{T}_{\mathrm{LL}})) \colon (s_1, s_2) \in \mathbf{R}.$$

Данное условие требует, чтобы при любом попадании в состояние абстрактной модели в отношении присутствовало соответствующее ему состояние детальной модели, в которое

детальная модель попала одновременно или ранее, чем случился данный переход в абстрактной модели.

Используя введённые термины, сформулируем следующий метод доказательства соответствия между разноуровневыми моделями в предположении, что верификация моделей **LTS1** и **LTS2** уже выполнена.

***Шаг* 1.** Определяется **R** – отношение соответствия между состояниями моделей **LTS1** и **LTS2**. Это можно сделать, используя термины только моделей **LTS1** и **LTS2**, то есть в виде машиночитаемой спецификации.

***Шаг* 2.** Заданное отношение **R** умозрительно проверяется на выполнение условие из определения 7. Проверка не может быть автоматизирована без построения модели событий низкого уровня.

***Шаг* 3**. Используя инструментальные средства автоматического или интерактивного доказательства доказывается утверждение:

$\forall (s_1, s_2) \in \mathbf{R}\ s_1 \in S_{\text{unsafe1}} \Rightarrow s_2 \in S_{\text{unsafe2}}$.

Если все шаги выполнены успешно, то можно сделать вывод, что целевые системы, описываемые моделью **LTS2**, удовлетворяют свойствам безопасности, доказанным для модели **LTS1**, т.е. $\mathbf{LLSafeT}_{\text{LTS2}} \subseteq \mathbf{LLSafeS}_{\text{LTS1}}$.

Доказательство.

Рассмотрим трассу $\mathbf{T}_{\text{LL}} \in \mathbf{LLSafeT}_{\text{LTS2}}$ и покажем, что тогда она принадлежит и $\mathbf{LLSafeS}_{\text{LTS1}}$. Предположим обратное, тогда найдётся $\mathbf{T'}_{\text{LL}} \in \mathbf{Prefix}^{\text{fin}}(\mathbf{T}_{\text{LL}})$, такой что $\mathbf{states}(\alpha_1(\mathbf{T'}_{\text{LL}})) \cap S_{\text{unsafe1}} \neq \emptyset$.

Обозначим s_k первое состояние в трассе $\alpha_1(\mathbf{T'}_{\text{LL}})$, такое что $s_k \in S_{\text{unsafe1}}$. Отметим, что s_k не может быть первым состоянием в трассе, т.к. тогда невозможно выполнить верификацию LTS1.

В силу конечности трассы $\mathbf{T'}_{\text{LL}}$ найдётся такой $\mathbf{T''}_{\text{LL}} \in \mathbf{Prefix}(\mathbf{T'}_{\text{LL}})$, что $s_k \in \mathbf{newstates}(\mathbf{T''}_{\text{LL}}, \alpha_1)$.

Тогда вследствие свойств отношения **R** найдётся $s_j \in \mathbf{states}(\mathbf{T''}_{\text{LL}}, \alpha_2)$: $(s_k, s_j) \in \mathbf{R}$. И так как $s_k \in S_{\text{unsafe1}}$, то по утверждению, сформулированному в Шаге 3, $s_j \in S_{\text{unsafe2}}$, а следовательно, найден $\mathbf{T''}_{\text{LL}} \in \mathbf{Prefix}^{\text{fin}}(\mathbf{T}_{\text{LL}})$, такой что $\mathbf{states}(\alpha_2(\mathbf{T''}_{\text{LL}})) \cap S_{\text{unsafe2}} \neq \emptyset$. Значит, $\mathbf{T}_{\text{LL}} \notin \mathbf{LLSafeS}_{\text{LTS2}}$, и т. к. верификация **LTS2** уже выполнена, то $\mathbf{T}_{\text{LL}} \notin \mathbf{LLSafeT}_{\text{LTS2}}$, что противоречит исходному предположению. □

3.2 Доказательство соответствия абстрактной модели поведения

Доказательства того, что целевые системы, описываемые более детальной моделью, удовлетворяют свойствам безопасности, доказанным для абстрактной модели не всегда достаточно. Некоторые значимые свойства могут быть отражены в абстрактной модели поведения, но не быть представлены в её свойствах безопасности. Тогда может быть поставлена задача доказательства того, что целевые системы, описываемые более детальной моделью, также соответствовали описанию корректного поведения абстрактной модели. Формально это можно сформулировать так: Если целевая система удовлетворяет требованию $\mathbf{LLSafeT}_{\text{LTS2}}$, то она удовлетворяет и требованию $\mathbf{LLSafeT}_{\text{LTS1}}$, или $\mathbf{LLSafeT}_{\text{LTS2}} \subseteq \mathbf{LLSafeT}_{\text{LTS1}}$.

В качестве первого подхода к решению данной задачи заметим, что если $\mathbf{LLSafeT}_{\text{LTS1}} = \mathbf{LLSafeS}_{\text{LTS1}}$, то из доказательства $\mathbf{LLSafeT}_{\text{LTS2}} \subseteq \mathbf{LLSafeS}_{\text{LTS1}}$ автоматически следует $\mathbf{LLSafeT}_{\text{LTS2}} \subseteq \mathbf{LLSafeT}_{\text{LTS1}}$.

Для выполнения равенства $\mathbf{LLSafeT}_{\text{LTS1}} = \mathbf{LLSafeS}_{\text{LTS1}}$ достаточно, чтобы

$\forall (s_1, \lambda_1, s'_1) \in \rightarrow_{\text{unsafe1}}\ \ s_1 \in S_{unsafe1} \vee s'_1 \in S_{unsafe1}$.

Если любой некорректный переход начинается в небезопасном состоянии или ведёт в небезопасное состояние, то любая трасса, не попадающая в $\mathbf{LLSafeT}_{\text{LTS1}}$, также не попадёт в $\mathbf{LLSafeS}_{\text{LTS1}}$. Поэтому если можно проверить модель **LTS1** на наличие этого свойства и если оно выполнено, то все свойства модели поведения отражены в её свойствах безопасности и дополнительных действий не требуется. Если это не так, то одним из способов решения поставленной задачи является доработка модели **LTS1**, чтобы это свойство стало выполняться.

Определение 8. Отношением соответствия между некорректными переходами моделей **LTS1** и **LTS2** будем называть отношение $\mathbf{RT} \subseteq \rightarrow_{\text{unsafe1}} \times \rightarrow_{\text{unsafe2}}$, которое удовлетворяет следующему условию:

$\forall\, \mathbf{T}_{\text{LL}} \in \mathbf{LLTraces}^{fin}\ \forall (s_1, \lambda_1, s'_1) \in \mathbf{newtrace}(\mathbf{T}_{\text{LL}}, \alpha_1) \cap \rightarrow_{\text{unsafe1}}$

$s_1 \in S_{\text{safe1}} \wedge s'_1 \in S_{\text{safe1}} \Rightarrow \exists (s_2, \lambda_2, s'_2) \in \alpha_2(\mathbf{T}_{\text{LL}})$: $\big((s_1, \lambda_1, s'_1), (s_2, \lambda_2, s'_2)\big) \in \mathbf{RT}$.

Данное условие требует, чтобы для любого некорректного перехода в абстрактной модели, начинающегося и завершающегося в безопасном состоянии, в отношении присутствовал соответствующий ему переход в детальной модели, который происходил одновременно или ранее, чем случился данный переход в абстрактной модели. Заметим, что когда $\forall(s_1, \lambda_1, s'_1) \in \rightarrow_{\text{unsaf}}\ \ s'_1 \in S_{\text{unsafe1}}$, тогда отношение RT может быть вырожденным (пустым).

Сформулируем второй метод доказательства соответствия между разноуровневыми моделями в предположении, что верификация моделей **LTS1** и **LTS2** уже выполнена, а также при помощи первого метода установлено соотношение $\mathbf{LLSafeT}_{\text{LTS2}} \subseteq \mathbf{LLSafeS}_{\text{LTS1}}$.

***Шаг* 1**. Определяется **RT** отношение соответствия между некорректными переходами моделей **LTS1** и **LTS2**. Это можно сделать, используя термины только моделей **LTS1** и **LTS2**, то есть в виде машиночитаемой спецификации.

***Шаг* 2.** Заданное отношение **RT** умозрительно проверяется на выполнение условие из определения 8. Проверка не может быть автоматизирована без построения модели событий низкого уровня.

***Шаг* 3**. Используя инструментальные средства автоматического или интерактивного доказательства доказывается утверждение:

$\forall \big((s_1, \lambda_1, s'_1), (s_2, \lambda_2, s'_2)\big) \in \mathbf{RT}\ (s_1, \lambda_1, s'_1) \in \rightarrow_{\text{unsafe1}} \wedge\ s_1 \in S_{\text{safe1}}\ \wedge\ s'_1 \in S_{\text{safe1}}$

$\Rightarrow (s_2, \lambda_2, s'_2) \in \rightarrow_{\text{unsafe2}} \vee\ s_2 \in S_{\text{unsafe2}} \vee s'_2 \in S_{\text{unsafe2}}$.

Если все шаги выполнены успешно, то можно сделать вывод, что целевые системы, описываемые моделью **LTS2**, также соответствуют описанию корректного поведения, заданного абстрактной моделью **LTS1**, т.е. $\mathbf{LLSafeT}_{\text{LTS2}} \subseteq \mathbf{LLSafeT}_{\text{LTS1}}$.

Доказательство.

Рассмотрим трассу $\mathbf{T}_{\text{LL}} \in \mathbf{LLSafeT}_{\text{LTS2}}$ и покажем, что тогда она принадлежит и $\mathbf{LLSafeT}_{\text{LTS1}}$. Предположим обратное, тогда найдётся $\mathbf{T'}_{\text{LL}} \in \mathbf{Prefix}^{\text{fin}}(\mathbf{T}_{\text{LL}})$, такой что $\alpha_1(\mathbf{T'}_{\text{LL}}) \cap \rightarrow_{\text{unsafe1}} \neq \emptyset$.

Обозначим (s_k, λ_k, s'_k) первый переход в трассе $\alpha_1(\mathbf{T'}_{\text{LL}})$, такой что $(s_k, \lambda_k, s'_k) \in \rightarrow_{\text{unsafe1}}$.

Если $s_k \in S_{\text{unsafe1}} \vee s'_k \in S_{\text{unsafe1}}$, то $\mathbf{T'}_{\text{LL}} \notin \mathbf{LLSafeS}_{\text{LTS1}}$, а следовательно, $\mathbf{T'}_{\text{LL}} \notin \mathbf{LLSafeT}_{\text{LTS2}}$, и, соответственно, $\mathbf{T}_{\text{LL}} \notin \mathbf{LLSafeT}_{\text{LTS2}}$, что противоречит предположению.

В силу конечности трассы $\mathbf{T'}_{\text{LL}}$ найдётся такой $\mathbf{T''}_{\text{LL}} \in \mathbf{Prefix}(\mathbf{T'}_{\text{LL}})$, что $(s_k, \lambda_k, s'_k) \in \mathbf{newtrace}(\mathbf{T}_{\text{LL}}, \alpha_1)$.

Тогда вследствие свойств отношения **RT** найдётся $(s_j, \lambda_j, s'_j) \in \alpha_2(\mathbf{T''}_{\text{LL}})$: $((s_k, \lambda_k, s'_k), (s_j, \lambda_j, s'_j)) \in \mathbf{RT}$. И так как $(s_k, \lambda_k, s'_k) \in \rightarrow_{\text{unsafe1}} \wedge\ s_k \in S_{\text{safe1}} \wedge\ s'_k \in S_{\text{safe1}}$, то по утверждению, сформулированному в Шаге 3, $(s_2, \lambda_2, s'_2) \in \rightarrow_{\text{unsafe2}} \vee\, s_2 \in S_{\text{unsafe2}} \vee\ s'_2 \in S_{\text{unsafe2}}$. А следовательно, найден $\mathbf{T''}_{\text{LL}} \in \mathbf{Prefix}^{\text{fin}}(\mathbf{T}_{\text{LL}})$, такой что $\mathbf{states}(\alpha_2(\mathbf{T''}_{\text{LL}})) \cap S_{\text{unsafe2}} \neq \emptyset$ или $\alpha_2(\mathbf{T''}_{\text{LL}}) \cap \rightarrow_{\text{unsafe2}} \neq \emptyset$.

Значит $\mathbf{T}_{\mathrm{LL}} \notin \mathbf{LLSafeT}_{\mathrm{LTS2}}$, что противоречит исходному предположению. □

4. Вопросы практического применения методов

Относительно традиционных подходов, основанных на отношении уточнения, которые обеспечивают установления более сложных соответствий между разноуровневыми моделями, вплоть до отношения бисимуляции, предложенные методы доказательства соответствия между разноуровневыми моделями нацелены на доказательство свойств безопасности в индивидуальной трассовой семантике, что позволяет сделать их более простыми и пригодными для применения при решении практических задач верификации сложных видов функциональных требований, таких как ограничения на возникающие информационные потоки в операционных системах ответственного применения.

Например, при верификации моделей управления доступом для операционных систем, построенных на основе ядра Linux [12, 13, 14], возникает три уровня моделей, для которых применение методов на основе отношения уточнения, оказывается проблематичным. Наиболее абстрактный уровень [15] соответствует семейству требований доверия к безопасности ADV_SPM «Моделирование политики безопасности» в терминологии стандарта ГОСТ Р ИСО/МЭК 15408 [16-18]. В нём описываются вопросы управления доступом и информационных потоков в абстрактных терминах пользователей, субъектов, объектов и т. д., которые не всегда прямо соответствуют терминам, в которых работает операционная система.

Следующий уровень соответствует семейству требований доверия к безопасности ADV_FSP «Функциональная спецификация» ГОСТ Р ИСО/МЭК 15408. В нём описываются требования к поведение целевой системы в терминах её интерфейса. В случае с операционными системами это может быть интерфейс системных вызовов, в котором фигурируют понятия процессов, файловых дескрипторов, сокетов и т. д.

Ещё один уровень возникает при моделировании поведения отдельных компонентов целевой системы, непосредственно реализующих требования безопасности. Например, в случае операционных систем это может быть компонент ядра, реализующих интерфейс LSM (Linux Security Module), в котором операции выполняются над внутренними структурами ядра, представляющими файл (inode) или запись о файле в директории (dentry).

Соответственно, предложенные методы могут быть использованы для доказательства соответствия между всеми видами моделей SPM-FSP, FSP-LSM и SPM-LSM.

4.1 Направления дальнейших исследований

Практические проблемы, которые при этом возникают, связаны с двумя моментами. Во-первых, традиционные языки формальных спецификаций нацелены на описание корректных моделей поведения, то есть для описания **LTS** в составе $(S, S_{\mathrm{init}}, L, \rightarrow_{\mathrm{safe}}, S_{\mathrm{safe}})$. Вопросы адекватной спецификации некорректных поведений $\rightarrow_{\mathrm{unsafe}}$ требуют дополнительного исследования. Вторым моментом, требующим анализа на практике, являются умозрительные проверки, выполняемые в рамках некоторых шагов предлагаемых методов. В то же время следует отметить, что явные и чёткие формулировки таких проверок являются предпочтительными по сравнению с достаточно распространённым подходом оставлять вопросы адекватности моделей своим неформализуемым прообразам за рамками рассмотрения.

Рассматривая вопрос спецификации некорректных поведений $\rightarrow_{\mathrm{unsafe}}$, в первую очередь, следует пояснить необходимость расширения для этих целей множества меток. Рассмотрим, например, модель целевой системы, в которой вызов функции умножения на два **dmul** представлен в множестве меток **L** событиями dmul(0,0), dmul(1,2), dmul(2,4) и т. д. Тогда если реализация этой функции для параметра 2 вернула 5 в модели не окажется метки dmul(2,5) для представления этого события. Таким образом, одним из шагов по описанию

спецификации некорректных поведений является расширение множества меток для представления таких некорректных событий. Во многих случаях это можно сделать путём автоматической трансформации исходной спецификации по некоторым правилам. Например, за счёт допуска нарушения ограничений на параметры событий. Тем не менее, это не всегда возможно, и тогда альтернативным решением может стать моделирование некорректным событий выделенной меткой **fail**. Но следует иметь в виду, что введение таких неестественных (аномальных) трансформаций может существенно усложнить умозрительные проверки требуемых свойств и тем самым повысить вероятность пропуска ошибки в ходе такой проверки.

Относительно спецификации некорректных переходов может быть рассмотрено два крайних взгляда. Если при интерпретации модели **LTS** в виде функции α состояние модели рассматривать как прямое отображение состояния реализации (взгляд белого ящика), то естественным шагом будет дополнение корректных переходов некорректными из каждого состояния в каждое с любой возможной меткой. Если же состояние модели рассматривать исключительно как модельное, формируемое по результатам предыдущих событий (взгляд чёрного ящика), то модель необходимо расширять, допуская появление всех возможных меток, а вот переходы определять в соответствии с её внутренней логикой поведения.

Ещё одним направлением для анализа является возможность совмещения предлагаемых методов с подходами динамической верификации и, в частности, тестирования на основе моделей. С одной стороны, при тестировании требуется решение задачи вынесения вердикта о корректности наблюдаемого поведения целевой системы, которое при наличии абстрактной модели реализуется через сбор трассы наблюдаемых событий, формирование на её основе абстрактных событий и соотнесения их с моделью. Поэтому при тестировании также актуален вопрос подхода к формированию функции α. С другой стороны, интересной возможностью может быть реализация автоматического проведения умозрительных проверок в ходе тестирования на тех трассах, которые будут возникать в ходе выполнения тестов.

4.2 Частичное соответствие между моделями

На практике также встречаются ситуации, когда абстрактная модель описывает более богатый спектр поведений, чем более детальная модель. Тогда более детальная модель не может соответствовать более абстрактной в соответствии с теми определениями, которые были введены выше. Тем не менее, потребность в доказательстве частичного соответствия между разноуровневыми моделями всё равно существует.

Для решения такой задачи предлагается предварительная трансформация абстрактной модели путём вычленения той функциональности, которая уже описана в детальной модели и дальнейшего применения предложенных методов относительно трансформированной абстрактной модели.

5. Заключение

В настоящей работе предложены методы доказательства соответствия между разноуровневыми моделями, нацеленные на доказательство свойств безопасности в индивидуальной трассовой семантике. Эти методы более просты и пригодны для применения при решении практических задач верификации сложных видов функциональных требований по сравнению с традиционными подходами, основанными на установлении отношения уточнения между моделями.

Для предложенных методов явно и чётко сформулированы умозрительные проверки, которые требуется выполнять вручную, поскольку их невозможно автоматизировать без построения модели низкоуровневых событий. Таким образом, методы позволяют хотя бы

ограниченным способом осуществить проверку адекватности моделей своим прообразам в физическом мире.

В качестве основного направления для дальнейших исследований рассматривается исследование вопросов практического применения предложенных методов, в частности, подходов к адекватной спецификации некорректных поведений, интеграции методов с существующими средствами спецификации и верификации формальных моделей, а также комбинация подхода с методами тестирования на основе моделей, в частности для частичной автоматизации проведения умозрительных проверок на тех трассах, которые будут возникать в ходе выполнения тестов.

Список литературы / References

[1]. B. Alpern, F.B. Schneider. Defining liveness. Information Processing Letters, vol. 21, issue 4, 1985, pp. 181-185.

[2]. A. Khoroshilov. On formalization of operating systems behaviour verification. In Proc. of 11th International Conference on Computer Science and Information Technologies (CSIT-2017), 2017, pp. 168-172.

[3]. В.В. Кулямин, Н.В. Пакулин, О.Л. Петренко, А.А. Сортов, А.В. Хорошилов. Формализация требований на практике. Препринт no. 13, ИСП РАН, 2006 г., 70 стр. / V.V. Kulyamin, N.V. Pakulin, O. L. Petrenko, A.A. Sortov, A.V. Khoroshilov. Formalization of requirements in practice. Preprint no. 13, ISP RAS, 2006, 70 pp. (in Russian).

[4]. J. He, C.A.R. Hoare, J.W. Sanders. Data refinement refined. Lecture Notes in Computer Science, vol. 213, 1986, pp. 187-196.

[5]. J.R. Abrial, D. Cansell, D. Méry. Refinement and reachability in Event-B. Lecture Notes in Computer Science, vol. 3455, 2005, pp. 222-241.

[6]. R.J.R. Back. Refinement of parallel and reactive programs. In M. Broy (ed). Program Design Calculi. Springer, 1993, pp. 73-92.

[7]. M. Butler. An approach to the design of distributed systems with B AMN. Lecture Notes in Computer Science, vol. 1212, 1997, pp. 223-241.

[8]. L. Aceto. Action Refinement in Process Algebras. Cambridge University Press, 1992, 283 p.

[9]. J. Derrick, E.A. Boiten. Non-atomic refinement in Z. Lecture Notes in Computer Science, vol. 1708, 1999, pp. 1477-1496.

[10]. G. Schellhorn. ASM refinement and generalizations of forward simulation in data refinement: A comparison. Theoretical Computer Sciences, vol. 336, issues 2-3, 2005, pp. 403–436.

[11]. П.Н. Девянин, В.В. Кулямин, А.Л. Оружейников, А.К. Петренко, А.В. Хорошилов, И.В. Щепетков. Способ верификации формальной автоматной модели поведения программной системы. Патент на изобретение №2682003, 2019 г. / P.N. Devyanin, V.V. Kulyamin, A.L. Oruzheinikov, A.K. Petrenko, A.V. Khoroshilov, I.V. Shchepetkov. A method for verifying a formal automaton model of a software system's behavior. Patent for invention No. 2682003, 2019 (in Russian).

[12]. П.Н. Девянин, Д.В. Ефремов, В.В. Кулямин, А.К. Петренко, А.В. Хорошилов, И.В. Щепетков. Моделирование и верификация политик безопасности управления доступом в операционных системах. М., Горячая линия – Телеком, 2019 г., 214 стр. / P.N. Devyanin, D.V. Efremov, V.V. Kulyamin, A.K. Petrenko, A.V. Khoroshilov, I.V. Shchepetkov. Modeling and verification of access control security policies in operating systems. M., Hotline – Telecom, 2019, 214 p. (in Russian).

[13]. П.Н. Девянин, В.В. Кулямин, А.К. Петренко, А.В. Хорошилов, И.В. Щепетков. Интеграция мандатного и ролевого управления доступом и мандатного контроля целостности в верифицированной иерархической модели безопасности операционной системы. Труды ИСП РАН, том 32, вып. 1, 2020 г., стр. 7-26. DOI: 10.15514/ISPRAS–2020–32(1)–1 / P.N. Devyanin, V.V. Kuliamin, A.K. Petrenko, A.V. Khoroshilov, I.V. Shchepetkov. Integrating RBAC, MIC, and MLS in Verified Hierarchical Security Model for Operating System. Programming and Computer Software, 2020, vol. 46, issue 7, 2020, pp. 443–453. DOI: 10.1134/S0361768820070026.

[14]. V. Kuliamin, A. Khoroshilov, D. Medvedev. Formal Modeling of Multi-Level Security and Integrity Control Implemented with SELinux. In Proc. of the 2019 International Conference on Actual Problems of Systems and Software Engineering (APSSE), 2019, pp. 131-136.

[15]. А.В. Хорошилов, И.В. Щепетков. ADV_SPM – Формальные модели политики безопасности на практике. Труды ИСП РАН, том 29, вып. 3, 2017 г., стр. 43-56 / A.V. Khoroshilov, I.V. Shchepetkov. ADV_SPM – Formal security policy models in practice. Trudy ISP RAN/Proc. ISP RAS, vol. 29, issue 3, 2017. pp. 43-56 (in Russian). DOI: 10.15514/ISPRAS-2017-29(3)-4.

[16]. ГОСТ Р ИСО/МЭК 15408-1-2012 Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 1. Введение и общая модель / ISO/IEC 15408-1:2012 Information technology - Security techniques - Evaluation criteria for IT security - Part 1: Introduction and general model.

[17]. ГОСТ Р ИСО/МЭК 15408-2-2013 Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 2. Функциональные компоненты безопасности / ISO/IEC 15408-2:2013 Information technology - Security techniques - Evaluation criteria for IT security - Part 2: Security functional requirements.

[18]. ГОСТ Р ИСО/МЭК 15408-3-2013 Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 3. Компоненты доверия к безопасности / ISO/IEC 15408-3:2013 Information technology - Security techniques - Evaluation criteria for IT security - Part 3: Security assurance requirements.

Информация об авторе / Information about the author

Алексей Владимирович ХОРОШИЛОВ – кандидат физико-математических наук, ведущий научный сотрудник, директор Центра верификации ОС Linux в ИСП РАН, доцент кафедр системного программирования МГУ, ВШЭ и МФТИ. Основные научные интересы: методы проектирования и разработки ответственных систем, формальные методы программной инженерии, методы верификации и валидации, тестирование на основе моделей, методы анализа требований, операционная система Linux.

Alexey Vladimirovich KHOROSHILOV – Ph.D. in Physics and Mathematics, Leading Researcher, Director of the Linux OS Verification Center at ISP RAS, Associate Professor of System Programming Departments at Moscow State University, the Higher School of Economics, and Moscow Institute of Physics and Technology. Main research interests: design and development methods for critical systems, formal methods of software engineering, verification and validation methods, model-based testing, requirements analysis methods, Linux operating system.